G3 CAN-J1939 configuration guide (v1.1)

Revision log:
V1.1>add support for data type BITS (replace BOOLEAN)

Filename :	iotasset.txt
Location :	\user

1. Introduction

   The file 'iotasset.txt' contains the assets configuration that is required by the CAN-J1939 program to acquire data from CAN-J1939 devices and also pre-process for downstream IoT clients.

2. IOT asset 'KEY,VALUE' general format

   Each IOT asset is defined by using a BLOCK of 'key, value' pairs (CSV format).
   There are four CAN bus key names that must be present for each IOT asset.
   These CAN bus key names are reserved and cannot be used for custom key names.

CAN bus KEYS	**Description**
TYPE	Define the type of CAN bus communication
CANPGN	Define the Parameter Group Number (PGN)
CANSPN	Define the Suspect Parameter Number (SPN)
CANDATA	Define the J1939 raw data parsing and data type conversion

   Custom keys can be freely defined but limited to eight custom keys.
   Each asset block must include the same set of custom keys.
   Backslash (\) and double quote mark (") char cannot be used.
   Comments can be inserted by using the hash (#) sign.

   To ease parsing of different types of assets, the asset blocks need to be located between the start and end of block markers.

CAN bus BLOCK MARKER	**Description**
CAN_START	Define the start of CAN bus assets
CAN_STOP	Define the end(stop) of CAN bus assets

3. IOT asset 'KEY,VALUE' setup information

TYPE, m

Argument	Value	Description
m	J1939	CAN-J1939 protocol

CANPGN, n

Argument	Format	Value	Description
n	Hex	eg. FEEE/F004[#1]	n = Parameter Group Number

#1 Refer to your CAN device user manual for supported PGN lists.

CANSPN, s

Argument	Format	Value	Description
s	-	0	S = Suspect Parameter Number (for future development)

CANDATA, t, u, v [, x, y] datatype: INTEGER, STRING, FLOAT

CANDATA, B.b, c, v [, x, y] datatype: BITS

Argument	Value	Description
t, u	t=Byte start u=Byte length	Position of starting byte (dec) Length of byte (dec)
B.b, c	B.b= (Byte_start).(bit_start) c=bits length	Position of starting Byte.bit (dec) Length of bits (dec)
v	Data Type	Data type as conversion from CAN bus raw data
x	Multiplier	Value = Value*Multiplier + Adder [#2]
y	Adder	Value = Value*Multiplier + Adder [#2]

#2 Optional: for Data Type INTEGER & FLOAT, **both** x & y arguments required when applied.

4. Data Type definitions for OBD/CAN bus

DATA TYPE BITS

v [Data Type]	c [Data Length (bits)]	Description
BITS	1-8	1-8 bits to unsigned integer[#3]

#3 Binary value parsed will be converted to integer value, eg 1110_2 will be reported as 14_{10}.

DATA TYPE INTEGER

v [Data Type]	u [Data Length (bytes)]	Description
UINT8	1	8-bit data to **8-bit unsigned integer**
SINT8		8-bit data to **8-bit signed integer**
UINT16HL	2	8-bit data pair to **16-bit unsigned integer**, big endian
UINT16LH		8-bit data pair to **16-bit unsigned integer**, little endian
SINT16HL		8-bit data pair to **16-bit signed integer**, big endian
SINT16LH		8-bit data pair to **16-bit signed integer**, little endian
UINT32HLhl	4	8-bit data quad to **32-bit unsigned integer**, big endian
UINT32hlHL		8-bit data quad to **32-bit unsigned integer**, Word – little endian, Byte – big endian
UINT32LHlh		8-bit data quad to **32-bit unsigned integer**, Word – big endian, Byte – little endian
UINT32lhLH		8-bit data quad to **32-bit unsigned integer**, little endian
SINT32HLhl		8-bit data quad to **32-bit signed integer**, big endian
SINT32hlHL		8-bit data quad to **32-bit signed integer**, Word – little endian, Byte – big endian
SINT32LHlh		8-bit data quad to **32-bit signed integer**, Word – big endian, Byte – little endian
SINT32lhLH		8-bit data quad to **32-bit signed integer**, little endian

DATA TYPE STRING

v [Data Type]	u [Data Length (bytes)]	Description
STRING8	8	Set of eight 8-bit data to **8 ASCII characters** (abcdefgh)
STRING8R		Set of eight 8-bit data to **8 ASCII characters**, reversed (hgfedcba)
STRING4	4	Set of four 8-bit data to **4 ASCII characters** (abcd)
STRING4R		Set of four 8-bit data to **4 ASCII characters**, reversed (dcba)

DATA TYPE FLOAT

v [Data Type]	u [Data Length (bytes)]	Description
FLOAT32ABCD	4	Set of four 8-bit data to IEEE-754 single precision floating point number. Byte orientation=ABCD,DCBA,BADC,CDAB A,B,C,D=canbyte1,canbyte2,canbyte3,canbyte4
FLOAT32DCBA		
FLOAT32BADC		
FLOAT32CDAB		

5. Example for IOT asset configuration

#iotasset example for CAN-J1939 protocol

```
CAN_START                          #start of CAN bus block

TYPE, J1939                        #CAN type=J1939
CANPGN, FEEE                       #SPN=0xFEEE
CANSPN, 0                          #SPN=0
CANDATA, 1, 1, UINT8, 1,-40        #byte start=1, byte length=1, data type=unsigned 8-bit
                                   #integer, value=value*1 - 40
Unit, degC                         #custom key1
Key, EngineCoolantTemp             #custom key2, max supported=8

TYPE, J1939
CANPGN, F004                       #SPN=0xF004
CANSPN, 0
CANDATA, 4, 2, UINT16HL, 0.25, 0   #data type=unsigned 16-bit integer, value=value*0.2 + 0
Unit, rpm
Key, EngineRPM

TYPE, J1939
CANPGN, FE6C                       #SPN=0xFE6C
CANSPN, 0
CANDATA, 7, 2, UINT16HL
Unit, kmh
Key, VehicleSpeed
```

```
TYPE, J1939
CANPGN, F003                #SPN=0xF003
CANSPN, 0
CANDATA, 6.1, 2, BITS       #Byte start=6 ,bit start=1, bits length=2
Unit, -
Key, SpeedLimitStatus

CAN_STOP                    #end of CAN bus block
```

6. Method to upload 'iotasset.txt' file to G3

-Upload the iotasset.txt file from your computer using the 'Upload iotasset.txt' button in the 'IoT Hardware' tab.

-Put the iotasset.txt file in \user folder of USB drive (with label 'FATBOX').
Plug the USB drive into G3 and click the 'Upload to FATBOX' button in the 'Management' tab.

-Use SCP/Putty console or WinSCP.

<EOF>